Modeling time table based tram traffic

Daniel Lückerath*, Oliver Ullrich, Ewald Speckenmeyer
{lueckerath|ullrich|esp}@informatik.uni-koeln.de
Institut für Informatik, Universität zu Köln
Pohligstraße 1, 50969 Köln, Germany

Abstract

In mid-sized cities, tram networks are a major component of the public service infrastructure. In those networks with their typically dense schedules multiple lines share tracks and stations, resulting in a dynamic system behavior and mounting delays following even small disruptions. Robustness is an important factor to keep delays from spreading through the network and to minimize average delays.

As part of a project on simulation and optimization of robust schedules, this paper describes the development, implementation and application of a simulation model representing a tram network and its assigned time table. We begin by describing the components of a tram network, which consist of physical and logical entities. These concepts are then integrated into a model of time table based tram traffic. We apply the resulting simulation software to our hometown Cologne's tram network and present some experimental results.

1 Introduction

Tram networks are important parts of the public transport infrastructure, in our hometown Cologne's tram network for example 745,000 passengers are transported every day [6]. Especially mid-sized cities often have mixed tram networks, i.e. networks where trams travel on street level (and are thus subject to individual traffic and corresponding traffic regulation strategies) and on underground tracks. In such dense networks robustness is an important factor to minimize average delays. Robustness measures the degree on which inevitable small disturbances, e.g. obstructed tracks due to parked cars, have impact on the whole network. With robust time tables delays are kept at a local level, whereas with non robust time tables they spread through the network and might subsequently cause delays of other vehicles [3, 4].

In this paper we present parts of a larger project to generate and evaluate robust time tables in order to minimize the impact of small delays, as written in [5]. We develop a model and implement an application to simulate time tables of mixed tram networks in order to evaluate given time tables before their implementation in the field and to compare time tables generated by optimization methods (as in [2]) in respect to their applicability. In addition we want to show that the adopted simulation engine can be applied to real world problems.

We begin the remainder of this paper by describing the basics of time table based tram traffic (section 2), followed by a short discussion of our model representing the physical

* The author was supported by the RheinEnergie Stiftung Jugend Beruf Wissenschaft under the grant number W-10-2-002.

and logical entities of the tram network (section 3). The resulting software is then applied to Cologne's tram network (section 4). We close with a short summary of the lessons learned and give some remarks on further research (section 5).

2 Time table based tram traffic

Tram networks can be considered as a combination of physical and logical components. The physical network consists of tangible entities, e.g. stations, tracks or trams, whereas the logical network is comprised of concepts and plans, e.g. lines, trips or time tables. Figure 1 shows an extract of an example network.

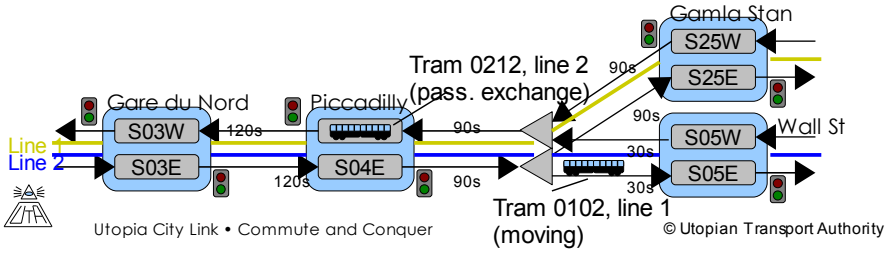

Figure 1: Part of a tram network

At the beginning of each *turn*, which is the planned movement of a tram through the network on a specific operational day, a tram leaves the *maintenance and storage depot* where it was stored over night. It then travels to the first *platform* of its first *trip*, where the passenger exchange takes place. Platforms are usually unidirectional and always part of a *station*, which combines adjacent platforms under a common name.

At certain stations and points in time *connection warrants* are in place. This means that vehicles located at different platforms of a station wait for each other, so that passengers have the chance to catch a follow-up connection. For example in the Cologne network connections are warranted between 23:00 and 01:00 at Neumarkt station every night.

After executing the passenger exchange the vehicle travels to the next platform of the trip. The order of platforms which have to be visited is defined by the *line route*. Different line routes can be combined under a common name, thus constituting a *line*. For

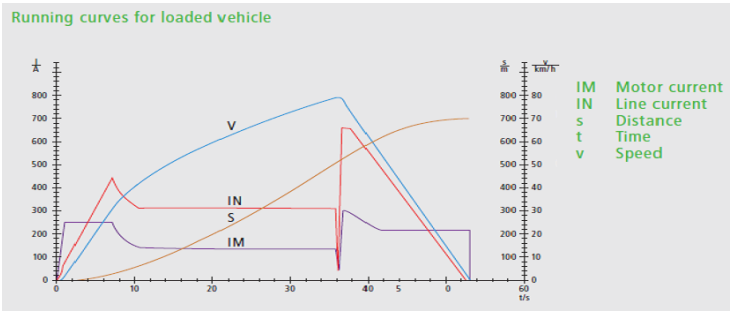

Figure 2: Maneuvering capabilities of wagon type K4000 as found in [9]

example Cologne's line 1 (from Junkersdorf to Bensberg and back) actually consists of 27 line routes, 15 of which are east bound and 12 are west bound.

The wagons used by the vehicle define the maneuvering capabilities and hence how the vehicle moves through the network. Table 1 depicts the most important characteristics for the three different wagon types which are in use in Cologne's tram network and figure 2 shows the maneuvering capabilities of wagon type K4000.

Characteristic	**K4000**	**K4500**	**K5000**
Length of wagon	29.2 m	29.0 m	29.3 m
Weight of wagon	35.0 t	39.0 t	37.8 t
Maximum velocity	80 kph	80 kph	80 kph
Acceleration	1.3 m/s²	-	1.2 m/s²
Normal brake ability	1.4 m/²	-	1.2 m/s²
Brake ability for emergency brake	3.0 m/s²	-	2.73 m/s²

Table 1: Characteristics of different wagon types as found in [9], [10] and [11]

The *tracks* between two locations of the network are usually unidirectional, but bidirectional tracks also exist. Some tracks may have speed limitations due to their environment, e.g. inner-city tracks may have a speed limit because of traffic regulations.

While the vehicle travels from one platform to another it may have to traverse *track switches*. These are locations where more than two tracks meet and they can be differentiated between dividing and joining track switches. Like platforms and tracks, track switches are usually unidirectional. All but one of the tracks sharing one side of the track switch must form a curve, which leads to speed limitations that are usually lower than the speed limits on tracks.

The access to track switches (as well as to platforms and track sections) is usually controlled by *traffic lights*, which switch between red and green phases of given length.

After the end of a trip a tram may need to turn around in order to start the next trip of its turn. Therefore *turning points* are situated between platforms that mark the end and beginning of line routes respectively. Usually these are stations with only one platform, where no passenger exchange takes place.

At the end of the operational day the vehicle travels once again to a maintenance and storage depot.

The spatial and chronological order of the vehicles in use on a specific operational day is constituted by the *time table*, i.e. the time table assigns each vehicle a turn and each turn a set of line routes with starting times.

3 Modeling tram traffic

3.1 Approach

Our approach to model and implement the described system is based on the characteristics of the adopted dynamic-adaptive parallel simulation engine, which is still under development and was up to now tested on randomized graphs only.

The framework follows a model-based parallelization approach, which tries to exploit the embedded model's intrinsic parallelism. To take maximum advantage of this, the engine is limited to systems that can be considered as sparse, directed graphs, which include many traffic simulation models.

While building the model a number of the applied simulation engine's requirements have to be met. Computation actually takes place in the nodes of the model graph. Each of those model nodes belongs at every instant to exactly one computational node, which can be a processor or processor core sharing a common cache with its neighbors or a remote computer connected via a network by message passing. Communication between those nodes takes place exclusively over the edges of the model graph. The means of communication are transparent to the model nodes. The simulation engine then takes care of dynamic load balancing, its mechanics are beyond the scope of this paper and are described in [7].

Partial models of different granularity are mapped in our approach as model nodes which administrate transient entities, which in turn are sent via the edges as data packets.

3.2 Physical network

The tram network is modeled as a directed graph with platforms, tracks and track switches represented by nodes. Every node administrates its currently hosted vehicles. Connections between nodes are represented as edges. Figure 3 depicts an example graph.

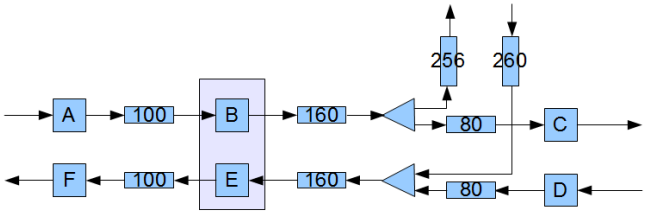

Figure 3: Example graph representing part of a tram network. Squares represent platforms, rectangles tracks and triangles track switches. The rectangle around platforms B and E indicates that these platforms form a station

At any point in time only one vehicle can be located at a platform. They are the main element for modeling boarding and deboarding of passengers and warranted connections. In the real world system passenger exchange is influenced by the platform and day time as well as tram type and passengers (e.g. number and speed). For the sake of simplicity we model the boarding and deboarding of passengers as loading time distributions specific to platform and tram type with the combined duration of opening and closing the vehicle doors as minimum value. In order to model the warranted connections each platform p holds a set of associated platforms, i.e. platforms whose trams wait for the tram at p and vice versa. Whenever a vehicle arrives at p during the warranted connections interval, p sends out a signal to all associated platforms. A vehicle which is located at a platform during the warranted connection interval stays at this platform until all associated platforms have sent a signal or a specific waiting time is exceeded. The

latter ensures that no infinite long waiting time occurs in case of cancellation of due vehicles.

Tracks are the only type of node that allows for more than one tram to be located at it at any point in time. The only exceptions to this rule are bidirectional tracks, which have to be exclusively reserved before a vehicle is allowed to enter them. In the real world system a bidirectional track could be passed by more than one tram traveling in the same direction, but to avoid infinite blocking of the opposite direction the model is simplified. Because the ÖPNV data model described in [8] does not allow for bidirectional connections between two locations of the network, they are modeled as two opposed unidirectional tracks. Reservation of one of the coupled tracks then causes blocking of the corresponding opposing track. Tracks also administrate traffic lights and blockings located on them.

As in [3] track switches are modeled as transfer points, i.e. they pass trams from an incoming to an outgoing track. Like platforms only one tram can use the track switch at any point of time. Hence they have to be reserved before being entered and unblocked afterwards. Track switches are the only node type that can have more than two neighbors.

As described above, traffic lights are administrated by tracks. Their position at the track is given as an offset related to the beginning of the corresponding track. Phase change is modeled as a function. This is possible because each traffic light has constant specific phase lengths t_{RED} and t_{GREEN}. Together with information about the initial state z_0 and the time of the first phase change t_0 the current status can be calculated as shown below.

$a : \mathbb{N}_0 \rightarrow \{0,1\}$, with *0 = Green* and *1 = Red*

$$a(t) = \begin{cases} 1 - z_{0,} & \text{if } 0 \le t - t_0 - j*(t_{\text{Green}} + t_{\text{Red}}) < t_{z_0} \\ z_{0,} & \text{else} \end{cases} \quad \text{with}$$

$$t_{\bar{z}_0} = \begin{cases} t_{\text{Green}}, & \text{if } z_0 = 1 \\ t_{\text{Red}}, & \text{if } z_0 = 0 \end{cases} \quad \text{and} \quad j = 0,1,\ldots,\left\lfloor \frac{\textit{maximum day time} - t_0}{t_{\text{Green}} + t_{\text{Red}}} \right\rfloor$$

The tram submodel contains most of the event based simulation logic, which is designed as described in [1]. Trams must always be located at a node of the network and their main attributes are specified by their type, which in turn is constituted by the type of wagons used. The tram type also holds functions for the maneuvering capabilities. As an example the velocity during acceleration from zero as a function of time for tram type K4000 is shown below.

$$v(t) = \begin{cases} 0 & \text{if } t < 1 \\ \frac{14}{3} * t - \frac{10}{3} & \text{if } 1 \le t \le 8 \\ 35{,}33223537 * \sqrt[3]{t} - 36{,}66447074 & \text{if } 8 \le t \le 36 \\ 80 & \text{else} \end{cases}$$

Additional tram types can easily be included in the model by extending the abstract base class. During the modeling process fourteen event types were identified (see table 2).

Trip start	Emergency brake start	Bidirectional track reservation
Trip end	Crash	Free bidirectional track
Tram standing	Passenger exchange start	Movement start
Acceleration start	Track switch reservation	Transfers to next node
Braking start	Free track switch	

Table 2: Identified events

As an example figure 4 shows the handling of event “tram standing” in pseudo code.

```
01 Event "tram standing" for tram t do
02      if t is located at a stop then
03              if passenger exchange completed then
04                      try to transfer t to next node
05                          (and if necessary reserve following bidirectional track)
06                      catch failed transfer by remaining to wait for n seconds
07              else execute passenger change
08      else if t is located on a track then
09              if t has reached end of track then
10                      try to transfer t to next node
11                          (and if necessary reserve following switch)
12                      catch failed transfer by remaining to wait for n seconds
13              else accelerate
```

Figure 4: Pseudo code algorithm for event type "tram standing"

3.3 Logical network

Most parts of the logical network do not have to be modeled explicitly, i.e. a line just combines a set of line routes under a common name and hence can be implemented as a simple string or integer value.

A line route on the other hand holds more information and therefore is modeled explicitly. Main component of a line route is a sorted list of platforms which have to be visited in this order. Because the ÖPNV data model contains no information about track switch locations on line routes, this information has to be computed prior to the simulation or dynamically before a tram tries to transfer to the next node. In order to identify individual line routes, each one is assigned a name and an unique ID.

Trips allocate a (planned) starting time to a specific line route and are assigned unique IDs. Each tram then holds a sorted list of trips, its turn. The set of turns of a specific operational day constitutes the time table of that day.

Blockings are the last logical network components which have to be modeled. They represent long term obstructions of tracks, e.g. traffic accidents or broken overhead lines, and make specific track positions impassable for a certain time.

3.4 Simulation infrastructure

In order to meet the requirements of the simulation engine the tram network is divided into disjunctive parts, each of which is then allocated to a model node. The special case of assigning the whole network to one model node results in a sequential simulation.

Each model node holds priority queues of blockings and trams located on the part of the network allocated to the node. If the model node receives the instruction to calculate the next simulation step it first inserts new vehicles, i.e. trams that were sent by neighboring model nodes, into the priority queue. It then instructs each blocking and vehicle whose time stamp is equal to the simulation time to execute the next simulation step. Finally all vehicles that need to be transferred are sent to neighboring model nodes.

To simplify the embedding of the tram network into the simulation engine and to decouple the simulation logic to the greatest possible extend from the communication and simulation engine, *transfer nodes* are inserted into the representation of the tram network. They assure the connection between adjacent parts of the network which are spread across different model nodes by holding information about destination model nodes, i.e. to which model node a tram has to be sent if it has to transfer.

Two types of transfer nodes exist: source and sink. *Sources* are entry points for trams that were sent to the model node at the end of the last simulation step and can only be found at the border of two adjacent parts of the tram network. *Sinks* represent end points and hold vehicles that have to be sent to neighboring model nodes at the end of the current simulation step.

4 Simulating Cologne's tram network

We apply the developed simulation software to our hometown Cologne's tram network based on the time table data of 2001. It consists of 528 platforms and 58 track switches connected via 584 tracks. These tracks cover a total length of 407.4 kilometers, resulting in an average track length of 697.6 meters. 15 lines with 182 line routes exist. On each operational day 2,814 trips are executed by 178 trams, transporting 745,000 passengers. [6]

We map each node of the graph representing the tram network as a model node, resulting in a run time of 392 seconds. Because the simulation engine does not yet support parallelization of the model (but the developed software uses the parallelization constructs) room for improvements regarding the run time exists.

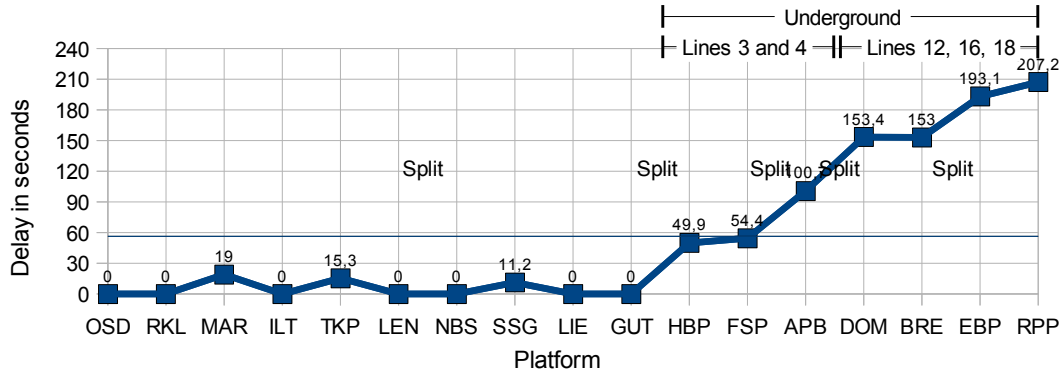

Figure 5: Line 5, Tram 507, Trip 3, starting at 7:07 at Ossendorf

For a first analysis we choose line 5 of the tram network. Serving 17 platforms it is the shortest line of the network and therefore best qualified for a detailed discussion. About half of the line runs through the inner city, while the other half runs through suburbs. It

shares most of its inner city tracks with lines 3, 4, 12, 16 and 18. Furthermore for about one third of its tracks line 5 travels underground.

Figure 5 depicts the average delay over the served platforms of trip no. 3 of tram 507, starting at 7:07 at Ossendorf station (OSD) and traveling to Reichenspergerplatz (RPP). During the first two thirds of its trip the tram travels along tracks not shared with other lines and thus no significant delay is accumulated. Because no vehicle leaves its current platform ahead of the planned departure time no travel time buffer is aggregated, as can be seen between Lenauplatz (LEN) and Nussbaumerstrasse (NBS), where the delay could be reduced well below zero.

After Gutenbergstrasse (GUT), where lines 3 and 4 join, the first significant delay occurs, because the tram has to coordinate with the other vehicles resulting in waiting times in front of track switches. These waiting times are generally higher than in the real world system because of a *first come first serve* approach for track switch reservation, resulting in possible blocking of faster but more distant vehicles. Furthermore the short inner city tracks prevent the vehicle from reducing the delay between platforms. The same pattern can be observed between Friesenplatz (FSP) and Dom/Hbf (DOM), where lines 3 and 4 separate from line 5 and lines 12, 16, and 18 join.

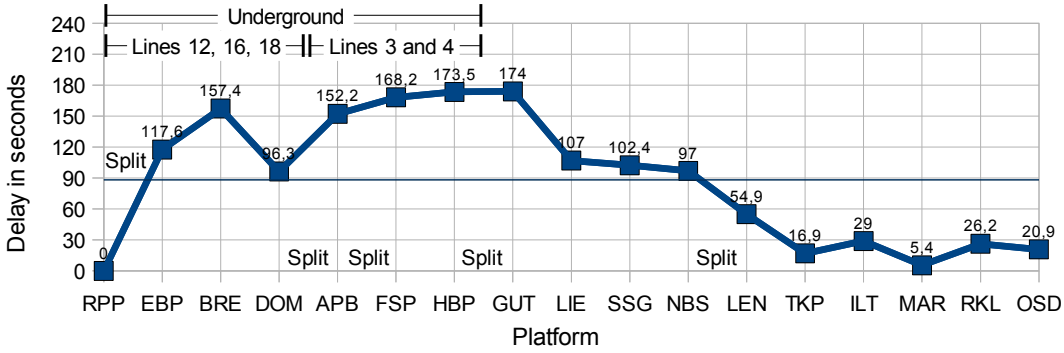

Figure 6: Line5, Tram 507, Trip 4, starting at 7:40 at Reichenspergerplatz

Figure 6 shows the follow-up trip of tram 507. Although trip no. 3 was finished with 207 seconds delay the next trip can start on time, because there is an adequate time buffer between these two trips. The increase of delay between RPP and Ebertplatz (EBP) in contrast to the more moderate one on the opposite direction can be explained by multiple track switches which have to be successfully reserved around Ebertplatz. From Breslauer Platz (BRE) to DOM the vehicle is able to reduce its delay by nearly 60 seconds, while in the opposite direction no such effect can be observed. The cause of this is that the planned travel time, which is measured from departure at one station to departure at the next, from BRE to DOM is 60 seconds higher than the travel time for the opposite direction, accounting for a higher expected time for passenger exchange at Dom/Hbf, which is a major national railway node. Because our model currently does not account for this the simulated vehicle is able to reduce the delay.

While the tram could not reduce the delay below zero between LEN and NBS during trip no. 3, it now uses the full potential of the comparatively long, planned travel time on

that track to reduce the delay by 42 seconds. This is also true for the track between Lenauplatz and Takuplatz (TKP).

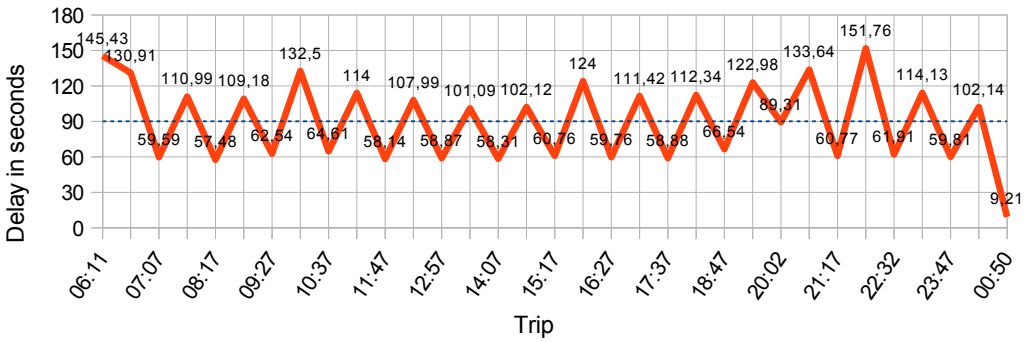

Figure 7: Average delay of trips of tram 507

Observing a tram over a whole operational day (tram 507, figure 7) we see a clear pattern: every trip from RPP to OSD has a higher average delay than its corresponding trip from OSD to RPP. The only exception to this is the first trip of the operational day when the tram moves from Subbelratherstrasse (SSG) to Reichenspergerplatz. During this trip the vehicle travels almost entirely along inner city tracks and subsequently the low delay suburban tracks have no impact. This over utilizes the time buffer between the trips and so results in a higher starting delay for the second trip, during which the delay cannot be reduced as much as during the following trips. The remaining trips are decoupled by significant time intervals.

The average delay of trips from OSD to RPP is lower than the average delay of the opposite direction, because vehicles traveling from RPP to OSD accumulate a very high delay over the first eight platforms and are not able to reduce it until GUT and even afterwards the delay can only be reduced significantly between NBS and TKP. On the other hand during trips from OSD to RPP trams accumulate almost no delay until Hans-Böckler-Platz (HBP), thus resulting in a much lower average delay.

5 Conclusion and future work

In this paper we described our approach for modeling time table based tram traffic. Beginning with a description of the structure of tram networks, which can be considered as a combination of physical and logical components, we described the different entities, e.g. trams, tracks or traffic lights, and their interaction.

After that we characterized our approach for modeling tram networks as graphs with trams as transient entities encapsulating most of the event based simulation logic, using the parallelization framework.

Finally we applied the developed simulation software to Cologne's tram network and analyzed some results. We were able to demonstrate that our application shows the expected behavior and although the delay in our example is higher than in the real world system, the results reflect the phenomena observable in Cologne's tram network. We also proved real world applicability of the simulation engine.

In further steps the developed model should be validated more closely, especially regarding the vehicle behavior at track switches. Then it should be applied to other generated as well as real world time tables for further evaluation.

6 References

[1] *Banks, J., Carson, J.S., Nelson B.L., Nicol D.M.*: Discrete-Event System Simulation, Upper Saddle River: Pearson, 2010.

[2] *Li, N., Speckenmeyer, E., Lückerath, D., Ullrich, O.*: Socio-economic Objectives in Metro Scheduling (Technical Report), Zentrum für angewandte Informatik Köln, 2011.

[3] *Lückemeyer, G.:* A Traffic Simulation System Increasing the Efficiency of Schedule Design for Public Transport Systems Based on Scarce Data. Aachen: Shaker Verlag, 2007.

[4] *Lückemeyer, G., Speckenmeyer, E.*: Comparing Applicability of Two Simulation Models in Public Transport Simulation. In: Becker, M, Szczerbicka, H. (Ed.): ASIM 2006 – 19. Symposium Simulationstechnik. ASIM/Universität Hannover, 2006.

[5] *Lückerath, D.*: ParSiVaL – Entwurf und Entwicklung einer Anwendung zur parallelen Simulation von schienengebundenem Öffentlichen Personennahverkehr. Diplomarbeit, Universität zu Köln, 2011.

[6] *Oberbürgermeister der Stadt Köln*: Stadtentwicklung in Köln – Mobilitätsentwicklung in Köln bis 2025. Stadt Köln, 2008.

[7] *Ullrich, O.*: Eine dynamisch-adaptive Architektur zur modellbasierten Parallelisierung von Simulationsanwendungen. Work in progress, Universität zu Köln.

[8] *Verband Deutscher Verkehrsunternehmen eV*: VDV-Standardschnittstelle Liniennetz/Fahrplan. VDV-Schriften 452 (2008).

[9] *Vossloh Kiepe GmbH*: Elektrische Ausrüstung des Niederflur-Stadtbahnwagens K4000 der Kölner Verkehrs-Betriebe AG. Druckschrift 00KV7DE, 2003.

[10] *Vossloh Kiepe GmbH*: Elektrische Ausrüstung des Niederflur-Stadtbahnwagens K4500 für die Kölner Verkehrs-Betriebe AG. http://www.vossloh-kiepe.com/vkproduktordner.2008-05-14.1154367607/vkproduktordner.2008-06-30.8585393121/vkproduktordner.2008-05-15.5609169940/vkprodukt.2008-05-14.8640403215/vkprodukt_download accessed on 24.05.2011.

[11] *Vossloh Kiepe GmbH*: Elektrische Ausrüstung der Hochflur-Stadtbahnwagen K5000 der Kölner Verkehrs-Betriebe AG. http://www.vossloh-kiepe.com/vkproduktordner.2008-05-14.1154367607/vkproduktordner.2008-06-30.8585393121/vkproduktordner.2008-05-15.5609169940/vkprodukt.2008-06-04.1250026636/vkprodukt_download accessed on 24.05.2011.